

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s):	Rabindranath Dutta, Karthikeyan Ramamoorthy		
Assignee:	International Business Machines Corporation		
Title:	System and Process for Enhancing Method Calls of Special Purpose Object-Oriented Programming Languages to Have Security Attributes for Access Control		
Serial No.:	09/817,102	Filing Date:	March 26, 2001
Examiner:	Syed Zia	Group Art Unit:	2131
Docket No.:	AUS920010044US1	Customer No.	65362

November 12, 2008

FILED ELECTRONICALLY

APPEAL BRIEF UNDER 37 CFR § 41.37

Dear Sir:

Applicant submits this Appeal Brief pursuant to the Notice of Appeal filed in this case on July 17, 2008, and the Notice of Panel Decision mailed September 12, 2008. A Request for extension of time is filed herewith. The fee for this Appeal Brief is being paid electronically via the USPTO EFS. The Board is authorized to deduct any other amounts required for this appeal brief and to credit any amounts overpaid to Deposit Account No. 090447.

I. REAL PARTY IN INTEREST - 37 CFR § 41.37(c)(1)(i)

The real party in interest is the assignee, International Business Machines Corporation as named in the caption above and as evidenced by the assignment set forth at Reel 011719, Frame 0661.

II. RELATED APPEALS AND INTERFERENCES - 37 CFR § 41.37(c)(1)(ii)

Based on information and belief, there are no appeals or interferences that could directly affect or be directly affected by or have a bearing on the decision by the Board of Patent Appeals and Interferences in the pending appeal.

III. STATUS OF CLAIMS - 37 CFR § 41.37(c)(1)(iii)

Claims 1-38 are pending in the application. Claims 1-38 stand rejected. The rejection of claims 1-38 is appealed. Appendix “A” contains the full set of pending claims.

IV. STATUS OF AMENDMENTS - 37 CFR § 41.37(c)(1)(iv)

On July 2, 2007, Applicants filed a Response to Non-Final Office Action amending independent claims 1, 9, 12, 15, 16, 24, 27, 30, and 31 to recite “an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.”

V. SUMMARY OF CLAIMED SUBJECT MATTER - 37 CFR § 41.37(c)(1)(v)

Independent claim 1 recites limitations for a process for restricting access to an object-oriented method within a data processing system. The process comprises initiating a call to the object-oriented method from a requester; (Paragraph [0072]; Figure 10B; Process 1010), determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct; (Paragraph [0072]; Figure 10B; Process 1012) in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, performing an authorization process to determine whether the requester is authorized to invoke the object-oriented method; (Paragraph [0073]; Figure 10B; Process 1016) and in response to a determination that the requester is authorized to invoke the object-oriented method, invoking the object-oriented method; (Paragraph [0074]; Figure 10B; Process 1022) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]). Independent claim 9 recites limitations for a process for restricting access to an object-oriented method within a data processing system. The process comprises: editing a source code statement that defines the object-oriented method within a source code file (Paragraphs [0071-0072]; Figures 10A-10B); and modifying the source code file to include an enforcement construct (Paragraphs [0049, 0071-0072; Figures 10A-10B]), wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process (Paragraphs [0058-0059]; Figures 7A-7B), prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the

object-oriented method; (Paragraph [0073]; Figure 10B) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 12 recites limitations for a process for restricting access to an object-oriented method within a data processing system. The process comprises: compiling source code that defines the object-oriented method within a source code file; (Paragraphs [0053-0058]; Figures 5-6) and compiling source code that defines an enforcement construct, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process (Paragraphs [0053-0058; 0071-0072; Figures 5-6, 10A-B]; prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method (Paragraph [0073]; Figure 10B); wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 15 recites limitations for a process of restricting invocation of an object-oriented method within a data processing system, the process comprising: identifying the object-oriented method within a data structure; (Paragraph [0072]; Figure 10B; Process 1012) and associating an object-oriented enforcement construct with the object-oriented method, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process that is to be executed, (Paragraph [0073]; Figure 10B; Process 1016) prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method; (Paragraph [0073]; Figure 10B) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 16 recites limitations for a computer program product in a computer-readable medium for use within a data processing system for restricting access to an object-oriented method. The computer program product comprises: instructions for initiating a call to the object-oriented method from a requester; (Paragraphs [0053-0058; 0072]; Figures 5-6, 10B; Process 1010), instructions for determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct; instructions for performing, in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, an authorization process to determine whether the

requester is authorized to invoke the object-oriented method; Paragraphs [0053-0058; 0073]; Figures 5-6, 10B; Process 1016), and instructions for invoking, in response to a determination that the requester is authorized to invoke the object-oriented method, the object-oriented method; Paragraphs [0053-0058; 0074]; Figures 5-6 10B; Process 1020) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 24 recites limitations for a computer program product in a computer-readable medium for use within a data processing system to generate source code for restricting access to an object-oriented method, the computer program product comprising: instructions for editing a source code statement that defines the object-oriented method within a source code file; (Paragraphs [0053-0058; 0071-0072]; Figures 5-6 10A-10B) and instructions for modifying the source code file to include an enforcement construct, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process, (Paragraphs [0049, 0058-0059]; Figures 5-6, 7A-7B) prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method; (Paragraph [0073]; Figure 10B) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 27 recites limitations for a computer program product in a computer-readable medium for use within a data processing system to generate executable code for restricting access to an object-oriented method. The computer program product comprises: instructions for compiling source code that defines the object-oriented method within a source code file (Paragraphs [0053-0058]; Figures 5-6); and instructions for compiling source code that defines an enforcement construct, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process, (Paragraphs [0053-0058; 0071-0072; Figures 5-6, 10A-B] prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method; Paragraph [0073]; Figure 10B) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 30 recites limitations for a computer program product in a computer-readable medium for use within a data processing system for restricting invocation of an object-oriented method. The computer program product comprises: instructions for identifying the object-oriented method within a data structure; (Paragraphs [0053-0058; 0072; Figures 5-6, 10B) and instructions for associating an object-oriented enforcement construct with the object-oriented method, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process that is to be executed, (Paragraph [0053-0058, 0073]; Figures 5-6, 10B; Process 1016) prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method; (Paragraphs [0053-0058; 0073]; Figures 5-6, 10A-B) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

Independent claim 31 recites limitations for an apparatus for restricting access to an object-oriented method within a data processing system. The apparatus comprises: means for initiating a call to the object-oriented method from a requester; (Paragraphs [0032-0043; 0072]; Figures 2A-2B, 10B; Process 1010), means for determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct; means for performing, in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, an authorization process to determine whether the requester is authorized to invoke the object-oriented method; (Paragraphs [0032-0043; 0073]; Figures 2A-2B, 10B; Process 1016), and means for invoking, in response to a determination that the requester is authorized to invoke the object-oriented method, the object-oriented method; (Paragraphs [0032-0043; 0074]; Figures 2A-2B, 10B; Process 1020) wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium (Paragraph [0078]).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

In the Final Office Action dated April 17, 2008, the Examiner finally rejected Claims 1-38 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,044,373 to Gladney et al. (“Gladney”).

VII. ARGUMENTS

Applicants' invention relates to generating and using an enforcement construct within a special purpose object-oriented programming language in order to control access to a protected method. More specifically, as set forth in previously amended independent Claims 1, 9, 12, 15, 16, 24, 27, 30, and 31, "an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium." Accordingly, within source code statements, the enforcement construct comprises an enforcement keyword (the "instruction") that indicates an authorization restriction on the invocation of an object-oriented method in conjunction with an identifier of an authorization mechanism, such as an authorization method. In the runtime environment, when a call is initiated to an object-oriented method, a check is made as to whether the object-oriented method has been protected by an enforcement construct. If so, then the identifier of the associated authorization method is used to invoke the authorization method, which determines whether an entity that is attempting to call or invoke the object-oriented method is authorized to execute the object-oriented method. If so, then the object-oriented method is invoked, and if not, an error response may be returned to the calling entity. The enforcement construct may be applied at the class level such that each method defined within a class becomes a protected method.

The Gladney reference is directed to a system and method for controlling a client's access to a protected element, wherein the protected element is contained in a protected resource which includes a data manager. In Gladney, the protecting resource and the protected resource are arranged in a distributed configuration.

Applicants agree with Examiner that Gladney teaches a method and computer program product for object oriented access control. However, Applicants respectfully disagree with Examiner that Gladney teaches the application of an enforcement construct at the class level such that each method defined within a class becomes a protected method. While the authorization process of Gladney enforces access to objects and protected methods, authorization is enforced at an application level with the data manager making calls to the protecting resource manager. In contrast, the present invention enforces authorization at the object level.

As an example, a first code portion defines class "ClientClassA" that contains method "A" that invokes method "M" in class "ServerClassM" and a second code portion defines a class "ClientClassB" that contains method "B" that likewise invokes method "M" in class "ServerClassM." The first code portion, comprising class "ClientClassA," is compiled into a

first application or module while the second code portion, comprising class "ClientClassB," is compiled into a second application or module. The first application or module is used by a user with a first authority role and the second application or module is used by a user with a second authority role. At some point in time, the first user requests some type of action that causes the execution of method "A" within class "ClientClassA." Likewise, at some point in time, the second user requests some type of action that causes the execution of method "B" within class "ClientClassB." Both method "A" and method "B" will attempt to invoke method "M" of class "ServerClassM." When method "M" of class "ServerClassM" is invoked, an exception might be thrown.

More specifically, when method "A" and method "B" attempt to invoke method "M" of class "ServerClassM," it is not possible to predict whether protected method "M" will execute without reference to the state of the runtime environment. The request to invoke method "M" of class "ServerClassM" might result in the throwing of an exception, depending upon the result of the authorization mechanism, which relies upon predetermined authorization information. In this example, method "M" is protected by enforcing an authorization process defined by method "AuthMethod" of class "ManagerAuthorization." Hence, the invocation of method "M" within method "A" will successfully execute because method "M" was invoked on behalf of the first user, who has been authenticated as being one of a set of authorized users, as determined by the "ManagerAuthorization" mechanism. The invocation of method "M" within method "B" will not cause the execution of method "M" because method "M" was invoked on behalf of the second user, who has been determined by the "ManagerAuthorization" mechanism to not be one of a set of authorized users.

Applicants respectfully submit that this feature is not taught by Gladney, nor any other art of record. Applicants further submit that Gladney fails to provide a teaching of an object-oriented enforcement construct associated with the invocation of a method, as recited in the aforementioned independent claims.

In view of the foregoing, it is respectfully submitted that independent claims 1, 9, 12, 15, 16, 24, 27, 30, and 31 are allowable over the art of record. Furthermore, the pending dependent claims are allowable as being dependent upon allowable base claims.

VIII. CLAIMS APPENDIX - 37 CFR § 41.37(c)(1)(viii)

A copy of the pending claims involved in the appeal is attached as Appendix "A."

IX. EVIDENCE APPENDIX - 37 CFR § 41.37(c)(1)(ix)

None.

X. RELATED PROCEEDINGS APPENDIX - 37 CFR § 41.37(c)(1)(x)

There are no related proceedings.

XI. CONCLUSION

In view of the above arguments, it is respectfully urged that the rejection of the claims should not be sustained.

CERTIFICATE OF TRANSMISSION

I hereby certify that on November 12, 2008 this correspondence is being transmitted via the U.S. Patent & Trademark Office's electronic filing system.

/Gary W. Hamilton/

Respectfully submitted,

/Gary W. Hamilton/

Gary W. Hamilton
Attorney for Applicant(s)
Reg. No. 31,834

APPENDIX A - PENDING CLAIMS

1. (Previously Presented) A process for restricting access to an object-oriented method within a data processing system, the process comprising:
 - initiating a call to the object-oriented method from a requester;
 - determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct;
 - in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, performing an authorization process to determine whether the requester is authorized to invoke the object-oriented method; and
 - in response to a determination that the requester is authorized to invoke the object-oriented method, invoking the object-oriented method;wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.
2. (Original) The process of claim 1 further comprising:
 - in response to a determination that the requester is not authorized to invoke the object-oriented method, returning an error response to the requester for the call to the object-oriented method.
3. (Original) The process of claim 1 further comprising:
 - identifying the authorization process that is associated with the object-oriented method.
4. (Original) The process of claim 1 further comprising:
 - obtaining identity information associated with the requester; and
 - passing the identity information associated with the requester to the authorization process.
5. (Original) The process of claim 1 wherein the authorization process is performed by invoking an authorization method.

6. (Original) The process of claim 1 further comprising:
analyzing runtime environment information in order to determine whether an invocation
of the object-oriented method has been restricted with an object-oriented
enforcement construct.
7. (Original) The process of claim 1 wherein the invocation of the object-
oriented method has been restricted with an object-oriented enforcement construct applied at a
method level in a source code statement for the object-oriented method.
8. (Original) The process of claim 1 wherein the invocation of the object-
oriented method has been restricted with an object-oriented enforcement construct applied at a
class level in a source code statement for a class that includes the object-oriented method.
9. (Previously Presented) A process for restricting access to an object-
oriented method within a data processing system, the process comprising:
editing a source code statement that defines the object-oriented method within a source
code file; and
modifying the source code file to include an enforcement construct, wherein the
enforcement construct comprises an authorization process identifier associated
with an authorization process and a reserved word to be recognized by a compiler
as requiring runtime execution of the authorization process, prior to invoking the
object-oriented method, to determine whether an entity is authorized to invoke the
object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded
in compiled source code and said source code is stored in a computer-readable
medium.
10. (Original) The process of claim 9 wherein the enforcement construct is
included in the source code statement that defines the object-oriented method.
11. (Original) The process of claim 9 wherein the enforcement construct is
included in a source code statement that defines a class that includes the object-oriented method.

12. (Previously Presented) A process for restricting access to an object-oriented method within a data processing system, the process comprising:
compiling source code that defines the object-oriented method within a source code file;
and
compiling source code that defines an enforcement construct, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process, prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.
13. (Original) The process of claim 12 wherein the enforcement construct is included in a source code statement that defines the object-oriented method.
14. (Original) The process of claim 12 wherein the enforcement construct is included in a source code statement that defines a class that includes the object-oriented method.
15. (Previously Presented) A process of restricting invocation of an object-oriented method within a data processing system, the process comprising:
identifying the object-oriented method within a data structure; and
associating an object-oriented enforcement construct with the object-oriented method,
wherein the enforcement construct comprises an authorization process identifier associated with an authorization process that is to be executed, prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.

16. (Previously Presented) A computer program product in a computer-readable medium for use within a data processing system for restricting access to an object-oriented method, the computer program product comprising:
instructions for initiating a call to the object-oriented method from a requester;
instructions for determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct; instructions for performing, in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, an authorization process to determine whether the requester is authorized to invoke the object-oriented method; and
instructions for invoking, in response to a determination that the requester is authorized to invoke the object-oriented method, the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.
17. (Original) The computer program product of claim 16 further comprising:
instructions for returning, in response to a determination that the requester is not authorized to invoke the object-oriented method, an error response to the requester for the call to the object-oriented method.
18. (Original) The computer program product of claim 16 further comprising:
instructions for identifying the authorization process that is associated with the object-oriented method.
19. (Original) The computer program product of claim 16 further comprising:
instructions for obtaining identity information associated with the requester; and
instructions for passing the identity information associated with the requester to the authorization process.
20. (Original) The computer program product of claim 16 wherein the authorization process is performed by invoking an authorization method.

21. (Original) The computer program product of claim 16 further comprising:
instructions for analyzing runtime environment information in order to determine
whether an invocation of the object-oriented method has been restricted with an
object-oriented enforcement construct.
22. (Original) The computer program product of claim 16 wherein the invocation
of the object-oriented method has been restricted with an object-oriented enforcement construct
applied at a method level in a source code statement for the object-oriented method.
23. (Original) The computer program product of claim 16 wherein the invocation
of the object-oriented method has been restricted with an object-oriented enforcement construct
applied at a class level in a source code statement for a class that includes the object-oriented
method.
24. (Previously Presented) A computer program product in a computer-
readable medium for use within a data processing system to generate source code for restricting
access to an object-oriented method, the computer program product comprising:
instructions for editing a source code statement that defines the object-oriented method
within a source code file; and
instructions for modifying the source code file to include an enforcement construct,
wherein the enforcement construct comprises an authorization process identifier
associated with an authorization process and a reserved word to be recognized by
a compiler as requiring runtime execution of the authorization process, prior to
invoking the object-oriented method, to determine whether an entity is authorized
to invoke the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded
in compiled source code and said source code is stored in a computer-readable
medium.
25. (Original) The computer program product of claim 24 wherein the
enforcement construct is included in the source code statement that defines the object-oriented
method.

26. (Original) The computer program product of claim 24 wherein the enforcement construct is included in a source code statement that defines a class that includes the object-oriented method.

27. (Previously Presented) A computer program product in a computer-readable medium for use within a data processing system to generate executable code for restricting access to an object-oriented method, the computer program product comprising:
instructions for compiling source code that defines the object-oriented method within a source code file; and
instructions for compiling source code that defines an enforcement construct, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process and a reserved word to be recognized by a compiler as requiring runtime execution of the authorization process, prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.

28. (Original) The computer program product of claim 27 wherein the enforcement construct is included in a source code statement that defines the object-oriented method.

29. (Original) The computer program product of claim 27 wherein the enforcement construct is included in a source code statement that defines a class that includes the object-oriented method.

30. (Previously Presented) A computer program product in a computer-readable medium for use within a data processing system for restricting invocation of an object-oriented method, the computer program product comprising:
instructions for identifying the object-oriented method within a data structure; and

instructions for associating an object-oriented enforcement construct with the object-oriented method, wherein the enforcement construct comprises an authorization process identifier associated with an authorization process that is to be executed, prior to invoking the object-oriented method, to determine whether an entity is authorized to invoke the object-oriented method;

wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.

31. (Previously Presented) An apparatus for restricting access to an object-oriented method within a data processing system, the apparatus comprising:
means for initiating a call to the object-oriented method from a requester;
means for determining whether an invocation of the object-oriented method has been restricted with an object-oriented enforcement construct; means for performing, in response to a determination that access to the object-oriented method has been restricted with an object-oriented enforcement construct, an authorization process to determine whether the requester is authorized to invoke the object-oriented method; and
means for invoking, in response to a determination that the requester is authorized to invoke the object-oriented method, the object-oriented method;
wherein an instruction to enforce said object-oriented enforcement construct is embedded in compiled source code and said source code is stored in a computer-readable medium.

32. (Original) The apparatus of claim 31 further comprising:
means for returning, in response to a determination that the requester is not authorized to invoke the object-oriented method, an error response to the requester for the call to the object-oriented method.

33. (Original) The apparatus of claim 31 further comprising:
means for identifying the authorization process that is associated with the object-oriented method.

34. (Original) The apparatus of claim 31 further comprising:
means for obtaining identity information associated with the requester; and
means for passing the identity information associated with the requester to the
authorization process.

35. (Original) The apparatus of claim 31 wherein the authorization process is
performed by invoking an authorization method.

36. (Original) The apparatus of claim 31 further comprising:
means for analyzing runtime environment information in order to determine whether an
invocation of the object-oriented method has been restricted with an object-
oriented enforcement construct.

37. (Original) The apparatus of claim 31 wherein the invocation of the object-
oriented method has been restricted with an object-oriented enforcement construct applied at a
method level in a source code statement for the object-oriented method.

38. (Original) The apparatus of claim 31 wherein the invocation of the object-
oriented method has been restricted with an object-oriented enforcement construct applied at a
class level in a source code statement for a class that includes the object-oriented method.

EVIDENCE APPENDIX - 37 CFR § 41.37(c)(1)(ix)

None

RELATED PROCEEDINGS APPENDIX - 37 CFR § 41.37(c)(1)(x)

There are no related proceedings.